

# Deck-Building Based Kingdom Sim

- Modular structure
- Flexible tile-building system, featuring:
  - 12+ effect types
  - Contextual and conditional effect execution
  - Effects triggered by stage events
- Challenges and time modifiers
- Resource, influence, and balance systems

One thing I'm proud of is that I managed to create a visually appealing product (though not yet finished) using Midjourney, a UI asset pack\*, and a handful of stock icons.

\* <https://www.gamedevmarket.net/asset/rpg-and-mmo-ui-5-9913>



3



3



100

Stan królestwa: 66



100



Ekspansja (●)

- 40%



101



1

Rozwój

4/5



0

Obrona

0/5



2

Wiedza

1/5



Fatum

2/5



0

Nieprawość

1/5



0

Zagrożenie

0/5



0

Czynniki

brak czynników

Wyzwania

Poszukiwanie leku (0/12) | ⌚ 6



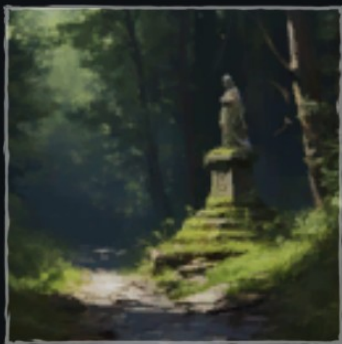
Q : ↺, E : 🔒

Decyzja 1/1 - Rozwój królestwa

👁 : 👁

↺ 3

Description label.



### Przydrożna kapliczka

Szansa na **+1** Wiedza: +10% za  
każdy sąsiadujący **wieśniak**  
lub **podróżny**.

No translation found for  
'CATEGORY\_POINTS\_CHANGE\_CH



### Zarządca wioski

Szansa na **+1** Rozwój lub  
Obrona -  
**+25%** za każdy kafelek na  
planszy, który jest **prosty** i  
**infrastruktura** lub **Wioska**

wieśniak



### Tawerna

Szansa na **Wojownik** - **20%**,  
**-5%** za każdy kafelek na  
planszy, który jest **bohater**  
Szansa na **+3** złoto za każdy  
kafelek na planszy, który jest

budynek

warsztat

Gameplay of the game can be viewed at the following address:

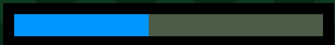
<https://drive.google.com/file/d/1wMFbjl8r3OGGL0QsehK8K7VBPDnf6eli/view>

Due to the absence of commentary, the pace of the gameplay (or rather, the walkthrough of the implemented mechanics) might feel a bit slow. I wanted to showcase all the mechanics I've implemented, but to avoid confusion, I opted for a slower approach - perhaps a bit too slow for some viewers. :)

## Base Building, Crafting Game

- Energy mechanics – energy required to perform tasks and being depleted in the process
- Processing mechanics – manual and automatic (self-resolving) nodes
- Containers mechanics – chests, equipment, moving items between them
- Crafting mechanics (workstations, processing, resources draining and generation points)
- Cultivation mechanics (including process modifications via applied fertilizers)
- Points of interest mechanics (framework for building interactions between player entity and interactable scene points / items)

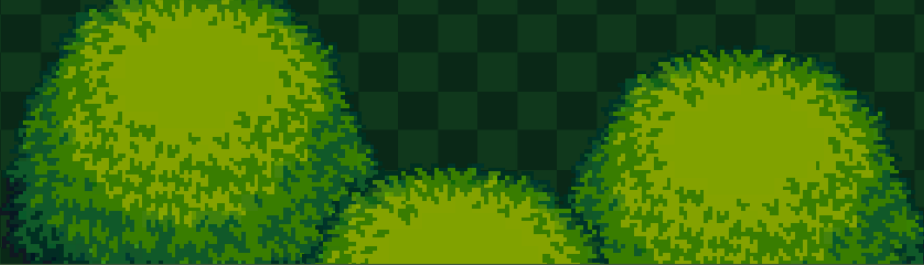




-6 ⚡



[F] Work



Crafting  
station

 0




Output:

Input:


Additional cost:




  
1

Bread

  
1 / 0

4 

00:03 



[E] C



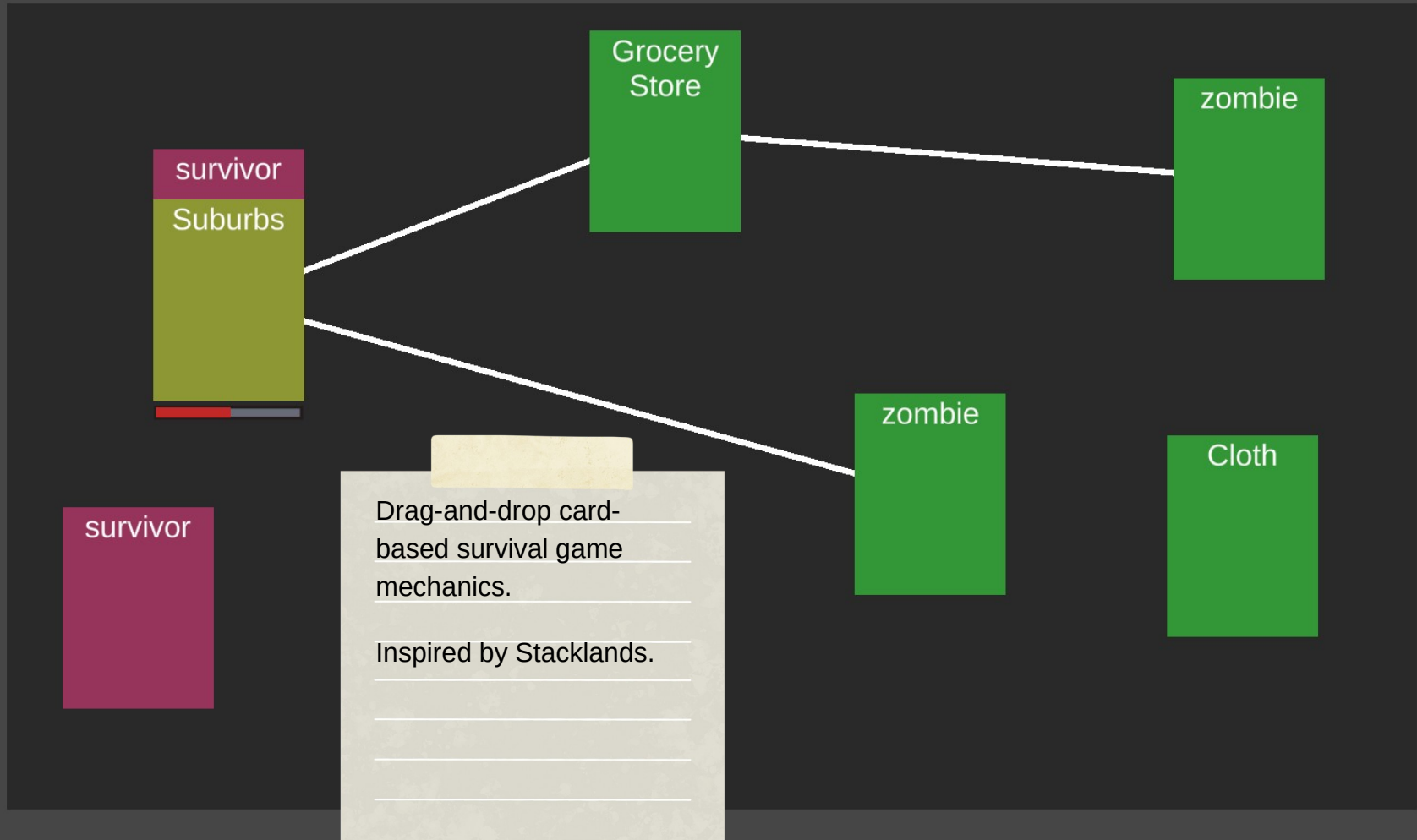
Gameplay of the game can be viewed at the following address:

<https://drive.google.com/file/d/1MEMTHwj70RLYiBpWcOQQdToL89owvAy4/view?usp=sharing>



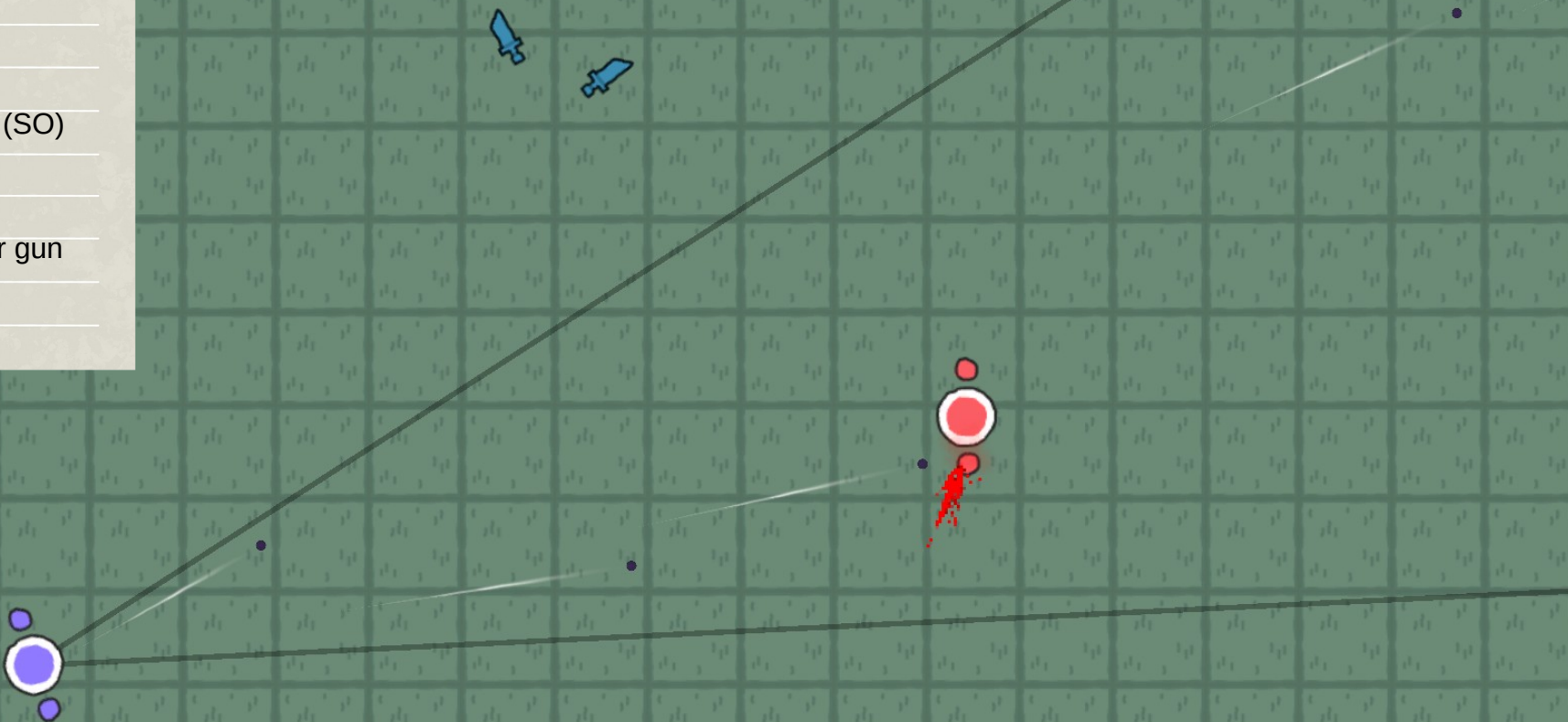
## Other Projects

Primarily smaller projects or implementations of specific features that piqued my interest.



## Top-Down Shooter

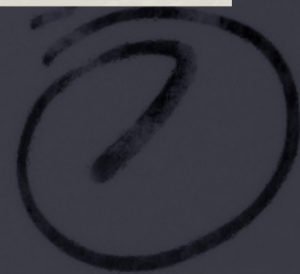
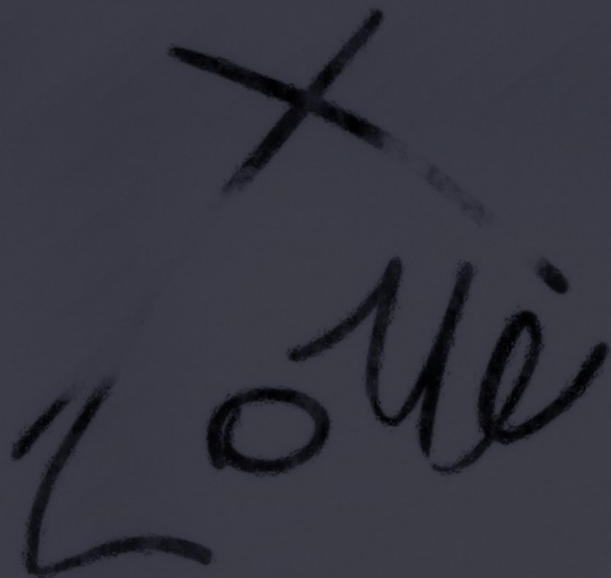
Testing the use of  
Scriptable Objects (SO)  
as instances of  
"strategies" in the  
process of modular gun  
building.





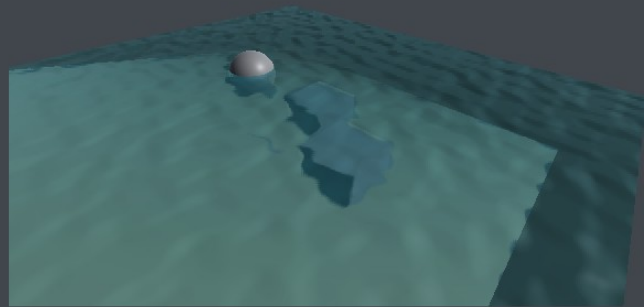
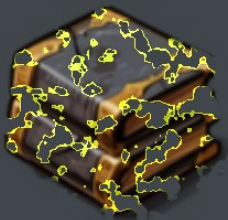
## Powerwash Mechanics

Utilizing custom  
materials created via  
Shader Graph and  
simple particle effect.



Further implementation  
of cleaning mechanics,  
including performance  
optimization, an  
animated brush, and  
support for multiple  
concurrent independent  
cleaning site instances.





Various shaders  
created using the  
Shader Graph.

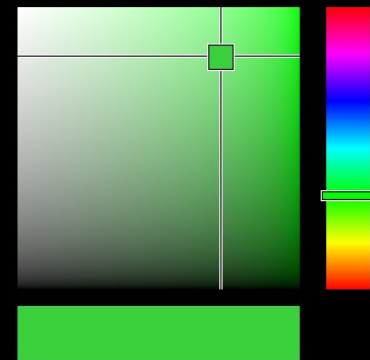


KA\_Idle01\_breathing\_Pose\_1

KA\_Idle05\_Stretch\_Pose\_1

Combat\_BareHands\_Combo\_Pose\_1

KA\_Combat\_BareHands\_Combo1



Color picker and mask-  
based texture segment  
coloring mechanics.

---

---

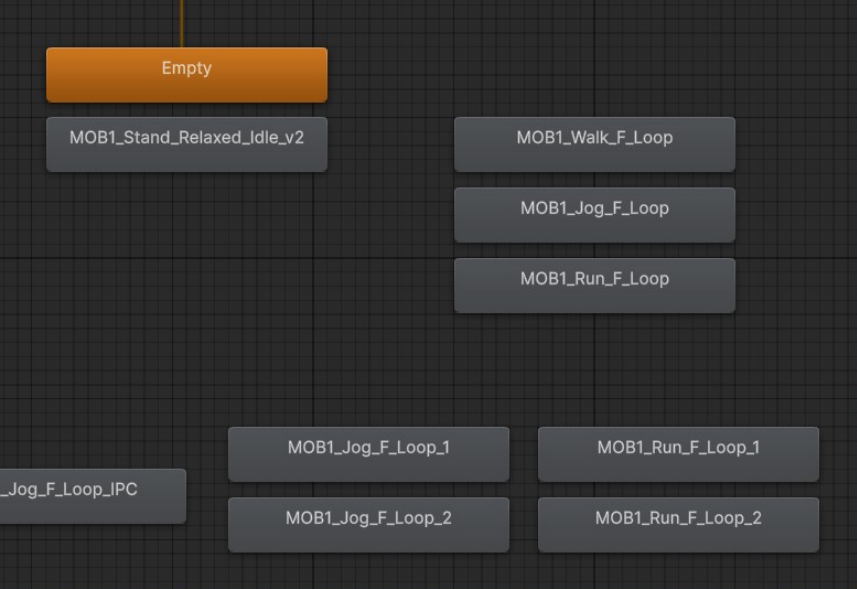
---

---

---

---





Branch  
[E] Pick up

```
[SerializeField]
AnimationEntriesFromInputTypeMap animationEntriesMap;
[SerializeField]
AnimationEntriesPriorityLayout animationEntriesPriorityLayout;
[SerializeField]
IndexFromAnimationLayerTagMap indexFromAnimationLayerTagMap;
[SerializeField]
float defaultCrossFadeDuration = 0.2f;

Animator animator;
Animation[] currentAnimations;

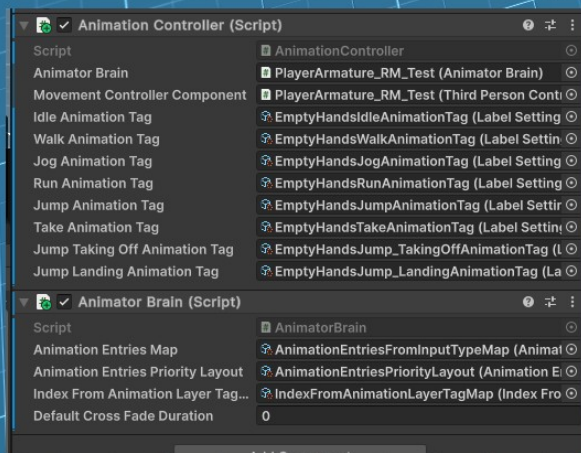
AnimationEntry currentAnimationEntry;

float motionTime = 0f;

Odwolania: 0
public Animator Animator => animator;

Odwolania: 3
protected override void ExtractArguments()
{
    animator = GetComponent<Animator>();
}

Odwolania: 5
public bool TryPlaying(LabelSettings animationTag)
{
    if (
        animationEntriesMap.TryGetMappedAnimationEntry(animationTag, out AnimationEntry targetA
        &&
        currentAnimationEntry != targetAnimationEntry
        &&
        animationEntriesPriorityLayout.CanBeOverriden(currentAnimationEntry, targetAnimationEntry))
    {
    }
```



Exploring a new approach to animating objects, utilizing a code-centric method instead of the Animator component.

*Created as a sample code project.*



Implementation of a simple framework for a tablet-based UI system (commonly used in VR applications). The repository is available for browsing at [https://bitbucket.org/meic\\_sample\\_projects/datapad\\_sample/src/main/](https://bitbucket.org/meic_sample_projects/datapad_sample/src/main/)